embedGAN: A Method to Embed Images in GAN Latent Space

Zhijia Chen, Weixin Huang(✉), and Ziniu Luo

School of Architecture, Tsinghua University, Haidian District, Beijing, China
huangwx@mail.tsinghua.edu.cn

Abstract. GAN is an efficient generative model. By performing a latent walk in GAN, the generation result can be adjusted. However, the latent walk cannot start from a selected image. The embedGAN is proposed to embed selected images into GAN and remain the generation effect. It contains an embedded network and a generative network. Application cases of residential interior design are given in the article. With advantages of a low computing cost and short training time, embedGAN shows its potential. The embedGAN algorithm framework can be applied to various GANs.

Keywords: embedGAN · GAN · Latent walk · Embedding data

1 Introduction

As a famous generative model in the field of machine learning, GAN has been widely used since proposed [1]. Further development of GAN has come up and one of the directions is to enable GAN to generate results according to users' control, for example, latent walks. The advantage of latent walks is that the pre-trained model is used and there is no need to retrain the model, which brings low computation cost and it is easy to check the effect. Latent walks can be applied to almost all types of GAN; and through the control of latent codes, directional generation can be achieved.

The problem of latent walks is that it must start from a randomly generated latent code, and cannot start from selected data. It is even unable to start from an image in the training data set, as GAN learns the sample features distribution, rather than the sample itself. At the same time, GAN training process also has the problem of insufficient diversity. To fundamentally solve the problem of generative diversity, it is necessary to expand the latent code dimension and make the training process invertible, like GLOW [9].

However, this approach requires large-scale computing power and a long training procession. If the goal is just to use any selected image for latent walk, is there a low-cost and rapid way to enable generating selected images in GAN?

The embedGAN method proposed in this article achieves this goal and its main functions are:

© The Author(s) 2021

1. Embed selected images in GAN generation distribution.
2. It hardly affects the parameters of other parts of the generative network, so that random walks in latent space can still be carried out.

The contributions of this research are:

1. A low-cost and effective method to expand the latent walk range of GAN is proposed.
2. The effectiveness of this method is verified by experiments, and examples applied to interior design are given.

2 Related Work

2.1 Regenerating Data in GAN

In the process of testing different evaluation standards, Shmelkov et al. tried to regenerate data in the training set through GAN, but found that images generated by GANs have a large gap with the selected ones [3]. A similar attempt was made in BEGAN and experiments were conducted to reverse the latent codes of real images. In most cases, GANs can hardly regenerate images from the training set [4]. Zhu et al. alleviated this problem by using multiple initial points and CNN to train the mapping of images to latent code [5]. Lipton et al. proposed a method to jump out of the local optimal solution [6]. An attempt is to train the images together with latent code. However, the results shows that the latent code still cannot generate the selected images well [7, 8]. It can be seen that generating selected data of the training set in GAN is an unsolved problem. The GLOW model can solve it, but it does not have the lightness of GAN, and it needs to pay a very high computing cost [9].

2.2 GAN Latent Walk

In DCGAN, parts of generative features are controlled by linear operations on latent code. But this attempt is limited to standardized data such as human faces [10]. In 3D-GAN, similar operations can also be achieved [11]. By introducing an aesthetic evaluator, Goetschalckx L et al. performed a heuristic search on the GAN latent space, and guided the generative network to produce images in the direction of higher aesthetic evaluation scores [12]. Jahanian A et al. succeeded in “steering” the GAN generative network by introducing an evaluator of lens shift [13]. Abdal et al. used a method of embedding images for StyleGAN, and studied the generation effect of various embedded images comprehensively [14]. However, the method is only applicable to StyleGAN. The method provided in this article can embed data in the initialized latent space and is widely applicable to various GANs.

3 Method

3.1 Principle

The input of GAN is a latent code and the output is an image. The input latent codes are usually generated from a multivariate Gaussian distribution, of which dimensions are lower than the real-world features. In the process of neural network training, due to the existence of ReLU and other activation functions, some feature information will be lost. For the reasons mentioned above, the discriminative network can hardly learn low-probability sample features, resulting in the generative network also unable to learn this part of the features or regenerate images from the training set (as the real images contains low-probability features) (Fig. 1).

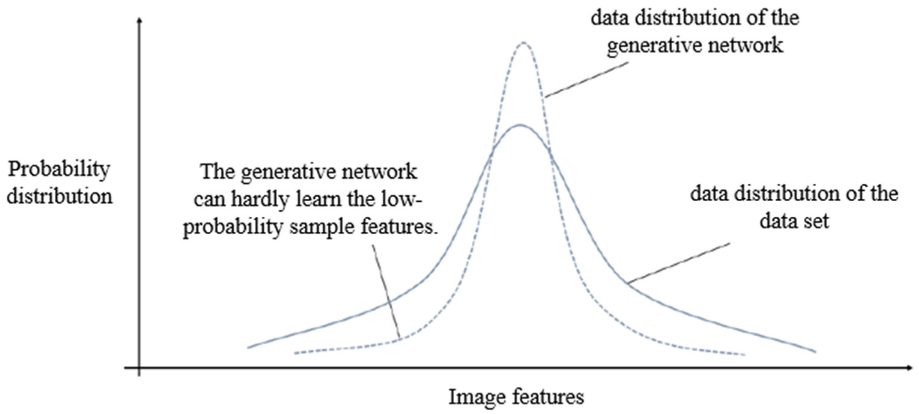

data distribution of the generative network
Probability distribution
The generative network can hardly learn the low-probability sample features.
data distribution of the data set
Image features

Fig. 1. The training set data distribution and the GAN generative data distribution.

In order to obtained selected images from the generative network, an idea is to fine-tune parameters of generative network and embed data. Through additional training, the generative network can be targeted to learn some low-probability features. This approach is equivalent to offsetting and expanding the data distribution, which improves the diversity of generated images (Fig. 2).

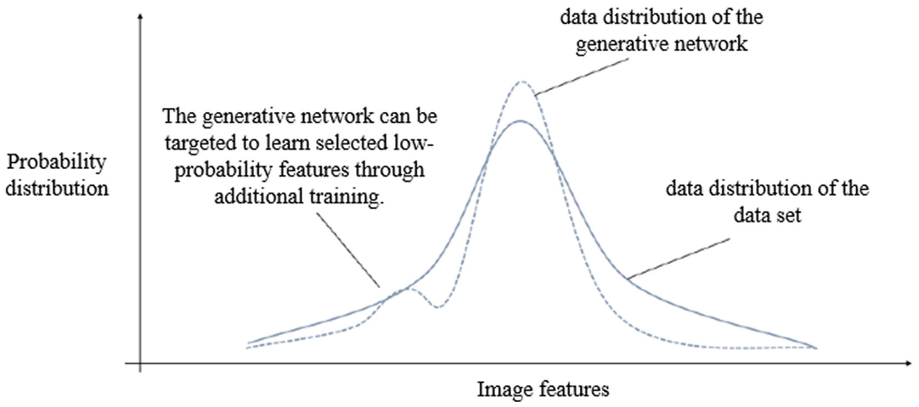
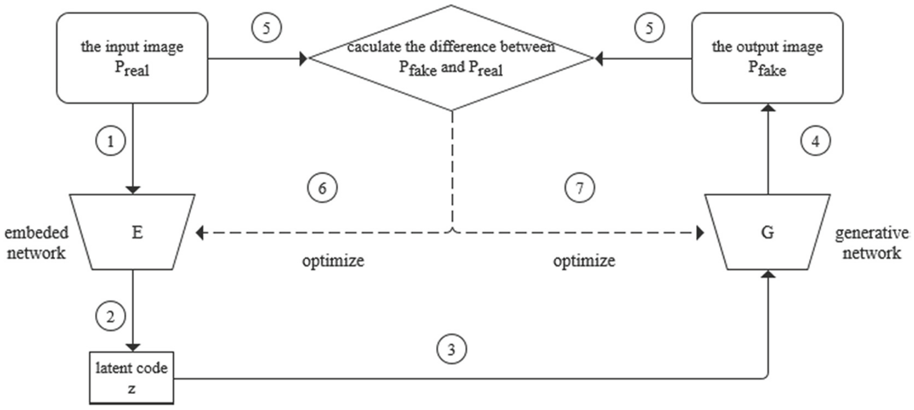

data distribution of the generative network
The generative network can be targeted to learn selected low-probability features through additional training.
Probability distribution
data distribution of the data set
Image features

Fig. 2. The training set data distribution and the GAN generative data distribution after embedding data

3.2 Architecture

Experiments show that if random codes are assigned to the selected images, the entire data distribution of the generative network will be distorted. Therefore, the key is to find an appropriate latent code for each selected image and minimize the impact on the distribution.

To achieve the above goals, a new GAN is designed, named embedGAN. EmbedGAN consists of a generative network and an embedded network. The two networks are mutually coordinated rather than antagonistic. To guarantee the basic quality of the output images, the generative network is from a pre-trained DCGAN. The embedded network consists of convolutional layers and fully connected layers. The embedded network outputs the appropriate latent codes according to the input target images, which also provides the direction of the updating parameter for the generative network. The architecture is shown in Fig. 3.



the input image P_{real}
5
caculate the difference between P_{fake} and P_{real}
5
the output image P_{fake}
1
4
6
7
embeded network
E
optimize
optimize
G
generative network
2
3
latent code z

Fig. 3. The structure of embedGAN. There are two parts of network training. Firstly, the real image P_{real} is input to the embedded network and the output is the code z. After z is input to the generative network, the embedded network is optimized according to the difference of P_{real} and the generative image P_{fake}. In the second round, the difference between P_{fake} and P_{real} is measured to optimize the generative network.

The optimization goal of embed GAN is

$$\min_{G,E} V(G, E) = [G(E(p_{real})) - p_{real}]^2$$

p_{real} refers to the selected real image data. $E(p_{real})$ refers to the latent code output by the embedded network. $G(E(p_{real}))$ refers to the image generated by $E(p_{real})$. The meaning of the formula is that the generation network G and the embedded network E should minimize the difference between the generated images and the selected real images.

3.3 Training Details

In the example, the data set consists 64 randomly selected real interior images. Before formal training, the embedded network can be pre-trained for 50 epochs. After pre-training, codes generated by the embedded network are input to the generative network, and it can be seen that there is still a large gap between the generated images and the target images (selected real images) (Fig. 4(a)(b)).

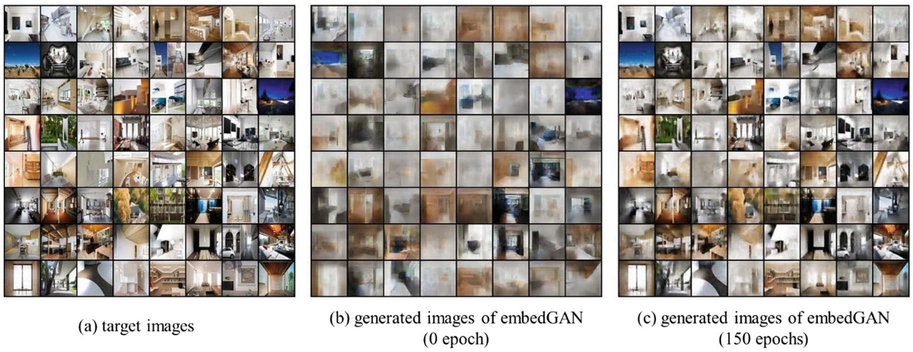

(a) target images (b) generated images of embedGAN (0 epoch) (c) generated images of embedGAN (150 epochs)

Fig. 4. Comparison of the target images and images generated by embedGAN.

In formal training, the learning rate of the generative network is 0.001 when the learning rate of the embedded network is 0.002. The loss function is MSE. The Adam optimization is used.

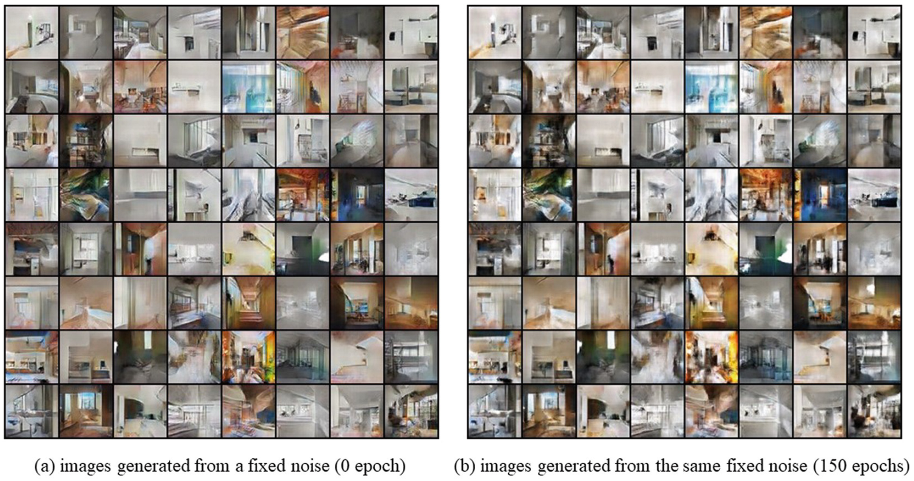

(a) images generated from a fixed noise (0 epoch) (b) images generated from the same fixed noise (150 epochs)

Fig. 5. Other images generated by a fixed random noise in embedGAN.

Images generated by embedGAN is shown as Fig. 4(c). It can be seen that after training, embedGAN does generate images closer to the target images. At the same time, images generated from other random codes have stayed high quality (Fig. 5). The training process of embed GAN only changes parts of the distribution, so that embedGAN can still generate rich interior images besides the selected images. EmbedGAN can be used to do latent walks with specific images.

4 Application

When designers need to explore various style features for existing residential projects, embedGAN can help explore the design intention. Randomly select the first image of "The Little White Apartment/Z-AXIS DESIGN" project (available on the website https://www.gooood.cn/little-white-apartment-z-axis.htm) from gooood for an application trial (Fig. 6). Limited by the training set, the size of the image is compressed to 64 * 64.

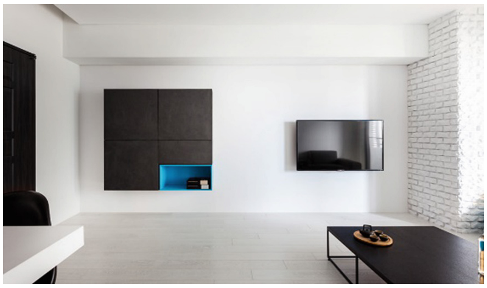


Fig. 6. Image from "The Little White Apartment/Z-AXIS DESIGN" project.

A classification network (consists of CNN layers and uses BCE as the loss function) is trained to discriminate the decorative materials used in interior spaces. Wood, concrete, metal stone, marble and veneer styles are classified and the accuracy of the testing set is 97%. Based on the classification network, we can perform a heuristic search in the latent space of embedGAN to regenerate certain features. The heuristic search is that when a certain material score of the image is increased, the result of this exploration is retained. The selected image is regenerated into different decorative material style (Fig. 7).

Similarly, complex features that are difficult to analyze intuitively can be generative. For example, by training a national discriminator of residential projects, different national styles can be regenerated (Fig. 8). As can be seen from the above applications, embedGAN achieves the ability to regenerate the selected real images into a selected feature direction.

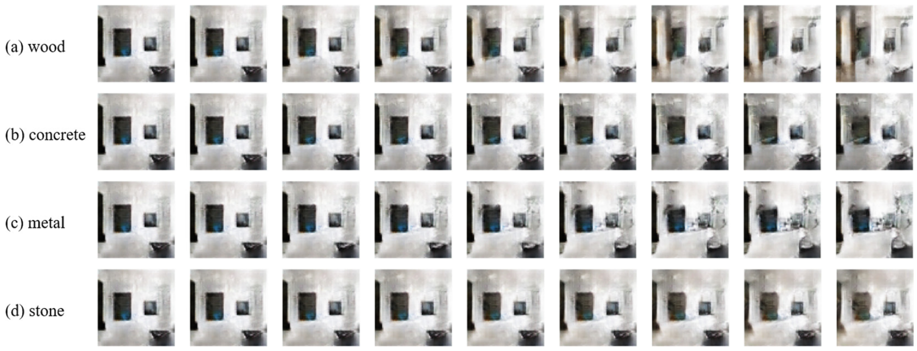


Fig. 7. Regenerated images in latent walks. From left to right, the generated image is more like the decorative style of the material

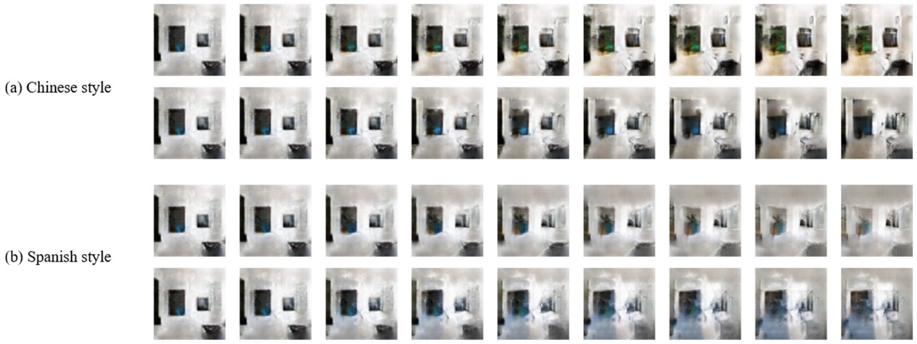


Fig. 8. Images regenerated based on the direction of nations. From left to right are generated images that are more in line with the country's style

5 Evaluation

The generation effect of embedGAN was evaluated in two aspects. On one hand, the difference between the images generated by embedGAN and the original images. On the other hand, the quality of randomly generated images.

The effects of image embedding are evaluated by MSE and SSIM. The RGB pixel mean square error (MSE) as well as structural similarity (SSIM) of the 64 embedded images and the 64 original selected images are calculated (Fig. 9). The average of MSE is 307.73 for the 64 * 64 pixel images, the mean deviation of each pixel is 0.07. The mean of SSIM is 0.72 (The closer to 1, the less different the two images are). It can be seen that embedGAN can generate images that have small differences from the selected images.

The embedGAN generative network and the original DCGAN generative network are compared to evaluate the random images generation quality. The two generative networks are tested by a same DCGAN discriminative network. It found that embedGAN has a larger loss function (Fig. 10). An explanation is that embedGAN is designed for a new problem (embedding images), which makes it works worse in the original problem

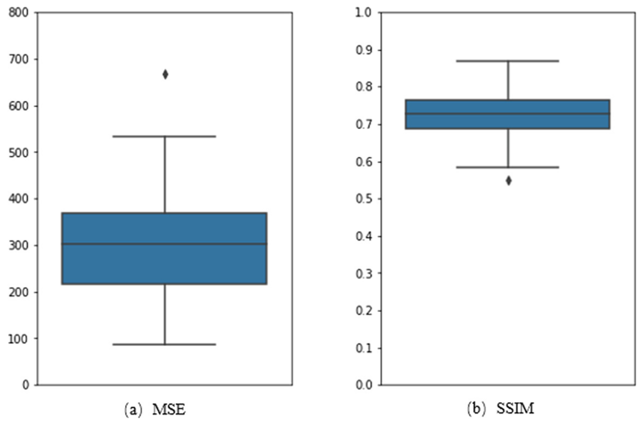


Fig. 9. The MSE and SSIM evaluation results.

(generating image from the original data set), according to the "there is no free lunch" principle. However, under the condition of generating design intent, embedGAN can still generate images closed enough to the data set. The loss of image quality is acceptable.

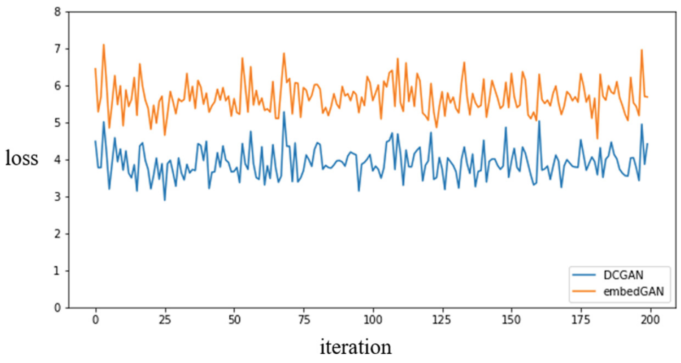


Fig. 10. Loss function of DCGAN and embedGAN generative network on a same DCGAN discriminative network.

6 Conclusion

This study is dedicated to solving the problem that when using GAN for latent walks, it is unable to start from a selected image. In the application of latent walks, embedding specific data points in parts of the generative data distribution is a method that takes both computation costs and image quality into account. This research solves this problem by proposing a framework that includes an embedded network and a generative network. This article verifies the effectiveness of this solution, and points out that after embedding the data, the quality of the generation has declined to some extents. However, under the condition of design intention, this decline is within an acceptable range. The method proposed in this paper is expected to be applied as a design tool.

References

1. Goodfellow, I.: NIPS 2016 tutorial: Generative adversarial networks. arXiv preprint arXiv:1701.00160 (2016)
2. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2019)
3. Shmelkov, K., Schmid, C., Alahari, K.: How good is my GAN?. In: Proceedings of the European Conference on Computer Vision (ECCV) (2018)
4. Berthelot, D., Schumm, T., Metz, L.: Began: boundary equilibrium generative adversarial networks. arXiv preprint arXiv:1703.10717 (2017)
5. Zhu, J.-Y., et al.: Generative visual manipulation on the natural image manifold. In: European Conference on Computer Vision. Springer, Cham (2016)
6. Lipton, Z.C., Tripathi, S.: Precise recovery of latent vectors from generative adversarial networks. arXiv preprint arXiv:1702.04782 (2017)
7. Donahue, J., Krähenbühl, P., Darrell, T.: Adversarial feature learning. arXiv preprint arXiv:1605.09782 (2016)
8. Dumoulin, V., et al.: Adversarially learned inference. arXiv preprint arXiv:1606.00704 (2016)
9. Kingma, D.P., Dhariwal, P.: Glow: generative flow with invertible 1×1 convolutions. In: Advances in Neural Information Processing Systems (2018)
10. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 (2015)
11. Wu, J., et al.: Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. In: Advances in Neural Information Processing Systems (2016)
12. Goetschalckx, L., Andonian, A., Oliva, A., et al.: Ganalyze: toward visual definitions of cognitive image properties. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 5744–5753 (2019)
13. Jahanian, A., Chai, L., Isola, P.: On the "steerability" of generative adversarial networks. arXiv preprint arXiv:1907.07171 (2019)
14. Abdal, R., Qin, Y., Wonka, P.: Image2StyleGAN: how to embed images into the StyleGAN latent space?. In: Proceedings of the IEEE International Conference on Computer Vision (2019)

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.